El harrachi

الجمهورية الجزائرية الديمقراطية الشعبية
People's Democratic and Republic of Algeria

Ministry of Higher Education and Scientific Research

National School of Cyber Security

وزارة التعليم العالي والبحث العلمي

المدرسة الوطنية العليا في الأمن السيبراني

Tutorials 1: Pointers

January 2025

Objectives: The goal of this series is to practice using pointers to manipulate arrays and strings. Students will learn how to use pointers to access, modify, and process arrays and strings efficiently.

Exercise 1: Pointers and Arrays

Objective: Learn how to use a pointer to access and modify an array.

1. Create an array of integers with 5 elements and initialize it with values.
2. Use a pointer to print all the elements of the array.
3. Modify the array elements using the pointer and print the array after modification.

Exercise 2: Dynamic Memory Allocation

Objective: Learn how to allocate memory dynamically with *malloc* and free the memory with *free.*

1. Dynamically allocate memory for an array of 10 integers and initialize it with some values.
2. Use a pointer to print the elements of the array.
3. Free the dynamically allocated memory after use.

Exercise 3: Inverting an Array with Pointers

Objective: Use pointers to invert an array.

1. Create an array of integers with 5 elements.
2. Use pointers to reverse the array.
3. Display the reversed array.

Exercise 4: Searching for an Element in an Array Using a Pointer

Objective: Use pointers to search for an element in an array.

1. Ask the user to enter an array of integers with n elements.
2. Use a pointer to search for a specific element in the array.
3. If the element is found, display its index.

Exercise 5: Searching for an Element in a Sorted Array (Binary Search)

Objective: Learn how to implement a binary search algorithm using pointers to search for an element in a sorted array.

1. Create a sorted array of integers.
2. Use a pointer-based approach to implement the binary search algorithm and search for a given element in the array.
3. If the element is found, print its index. If the element is not found, display a "not found" message.

Exercise 6: Sorting an Array Using Pointers (Bubble Sort)

Objective: Learn how to implement a sorting algorithm (bubble sort) using pointers.

1. Create an array of integers.
2. Implement the bubble sort algorithm using pointer arithmetic to swap the array elements.
3. Display the sorted array.

Exercise 7: Reversing an Array Using Pointers

Objective: Reverse the contents of an array using pointers.

1. Declare an array and initialize it with values.
2. Use pointers to reverse the array.
3. Display the reversed array.

Exercise 8: Pointers and Strings

Objective: Use pointers to manipulate strings.

1. Declare a character array for a string (e.g., `char str[20];`) and assign it a value.
2. Use a pointer to traverse the string and print each character.
3. Modify the string using the pointer.

Exercise 9: Concatenating Two Strings Using Pointers

Objective: Concatenate two strings using pointers.

1. Declare two character arrays (char str1[50], char str2[50]).
2. Use pointers to concatenate str2 to str1.

3. Display the concatenated string.

Exercise 10: Removing Spaces from a String Using Pointers

Objective: Remove all spaces from a string using pointers.

1. Declare a string char str[100].
2. Use pointers to traverse the string and remove all spaces.
3. Display the modified string (without spaces).

Exercise 11: Pointers and Structures

Objective: Use pointers to manipulate structures.

1. Declare a structure struct person with fields name (string) and age (integer).
2. Declare a pointer to struct person and assign it the address of an instance of the structure.
3. Use the pointer to access the structure's members and display the values.